MODELLING ADAPTIVE WEB APPLICATIONS

Irene Garrigós [1], Jaime Gómez [1] and Cristina Cachero [1]
IWAD Group
Departamento de Lenguajes y Sistemas Informáticos
Universidad de Alicante. SPAIN

ABSTRACT

Conceptual Modelling approaches for the web need extensions to specify dynamic personalization properties in order to design more powerful web applications. Current approaches provide techniques to support dynamic personalization, usually focused on implementation details. This article presents an extension of the OO-H conceptual modelling approach to address the particulars associated with the design and specification of dynamic personalization. The main benefit is that this specification can be modified without recompile the rest of the application modules. We describe how conventional navigation and presentation diagrams are influenced by personalization properties. In order to model the variable part of the interface logic OO-H has a personalization architecture that leans on a rule engine. Rules are defined based on a *User Model* and a *Reference Model*.

KEYWORDS

Web Engineering, Conceptual Modelling, Personalization, XML

1. INTRODUCTION

Current Web Engineering approaches [1] help designers to make easier the understanding, development, evolution and maintenance of web applications. These methods are based on new constructors and hypermedial views [2, 5] that broach the problem of navigation/presentation of the user across the information space. We argue that new techniques to extend metamodels with personalization aspects are needed. This article presents how the *Object Oriented Hypermedia Method (OO-H)* [4] is extended to support dynamic personalization. We describe how conventional navigation and presentation diagrams are influenced by personalization properties. These properties are captured in the form of external files written in XML, that represent the rules. These rules, that form the variable part of the interface logic, will be treated at execution time by an engine included in the execution architecture (see Fig.1).

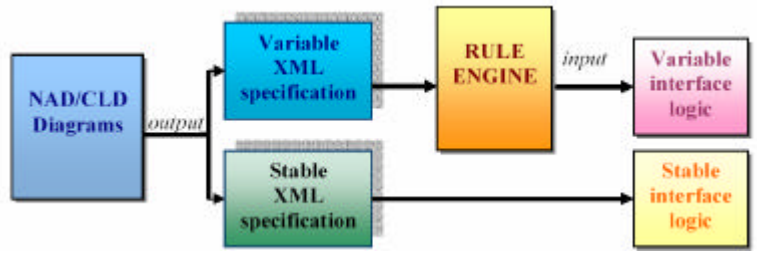

Figure 1. OO-H Personalization support

The remainder of the article is structured as follows: section 2 shows the elements that support the personalization in OO-H. Namely, subsection 2.1 presents how the modelling strategy of personalization is defined by means of a set of information structures expressed in a reference model. Subsection 2.2 describes, by means of a comprehensive example, how the dynamic personalization is supported in OO-H. Finally, section 3 presents the conclusions and further work.

2. EXTENDING OO-H TO SUPPORT DYNAMIC PERSONALIZATION

The architecture of OO-H has been extended to support dynamic personalization. More specifically, personalization properties are captured at navigation/presentation level and are reflected in their corresponding conceptual models (NAD, APD) by means of a set of association rules. In this way the design and the generation of the navigation logic is specified in two parts: an stable part, that is independent from the properties of personalization, and a variable part, that supports the treatment of these rules. Finally, a rule engine provides the context to interpret the generated rules at execution time. The support for this architecture is achieved by two models: A reference model, that also registers the user activity in the system. A user model that separates the personalizable part of the domain model from the invariable part. In the next section we will present an example that shows how dynamic personalization can be modelled in OO-H. The context is a video club in Internet that manages clients, movies and rentals.

2.1 Modelling Example

The *User Model* and the *Reference Model*, besides the *Association Rules*, let us to personalize the content, the navigation and the presentation of an application. In the context of the mentioned system (video club in Internet) we are going to consider the following requirement: *When a movie is rented, show recommendations of movies of the same category (Content Personalization).* First we are going to present the fundamentals of the OO-H Reference model.

2.1.1 The Reference Model

The OO-H Reference model contains the initial set of basic elements of information on which the desired personalization policy can be established. The main benefit the use of a reference model provides is that the designer can include and connect this framework with any model of the OO-H interface to which s/he wants to endow with the ability of personalization. The current OO-H reference model [3] structures the modelling of personalization in OO-H in three parts: user profiles, context information and association rules. For the purpose of this paper we must pay more attention to the part that stores the information of the association rules. OO-H incorporates an external mechanism to model the personalization of an application in form of association rules. The specification of the personalization in OO-H using association rules was presented in [3]. OO-H supports the specification of these rules by means of an XML schema. It is important to note that the execution architecture of the generated applications from OO-H models allows to modify and to reprocess this schema without recompiling the rest of modules. In the schema the different XML elements correspond to classes/attributes of the Reference model.

2.1.2 Support of the requirement

The first step to support this requirement is to define the *User Model*. In OO-H the *User Model* captures information about the features that the system believes the user has. The information that should contain this model depends on the personalization requirements that we want to support. In OO-H the user model is centred on the concepts of user and role. For the provision of personalized views to the user, OO-H provides a basic user model which is constructed around a class named *User*, that provides the information and behaviour that must be inherited by every *<<actor>>* of the system. This model can be enriched to support the desired personalization policy adding attributes, methods or links from the class User to the rest of the classes of the domain or the *OO-H Reference Model*. In this case one possible user model can be seen in the Fig. 3.

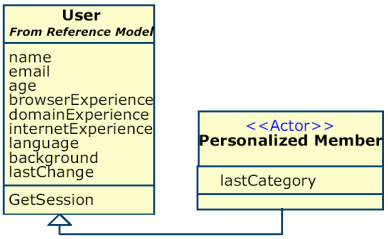

Figure 3. User model

This User model results from the connection of the *User* class, from the reference model, with a new class called *Personalized Member* that represents the variable part from the conceptual model. This new class is treated as a role. This class includes an attribute called *lastCategory* that stores the category of the last rented movie. To support this requirement we have to distinguish between the acquisition and personalization of the corresponding data. To specify each of the needed rules we are going to distinguish two phases: *Modelling and Gathering Process*, where it is shown how the requirement is introduced in the NAD diagram and *XML Specification* where it is shown an XML specification that supports the requirement.

Acquisition Rule (1a): MODELLING AND GATHERING PROCESS

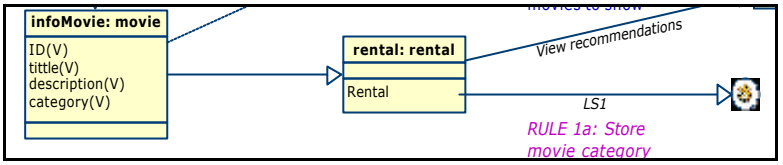

Figure 5. Acquisition: rule 1a

Figure 5 shows a portion of a NAD diagram in which the rule that models this requirement is introduced. The rule is called Store movie category. We can see that it is associated to a method invocation link. The method invocated is *Rent*, that is because we need to store the category of the rented movie to then show movie recommendations of the same category.

Acquisition Rule (1a): XML SPECIFICATION

The result of this acquisition process is stored as an XML specification that will be interpreted by a rule engine at execution time.

```
<TPersonalization>
    ...
    <profile name="ooh:all" condition="">
       <rule type="acquisition" name="StoreCategory" support="20"
             confidence="100" priority="Medium" activation="12/02/03"
             expires="12/02/04" lastTimeExecuted="23/09/03">
           <event type="MethodInvocation" link="Rent"/>
           <action value="session.member.lastcategory=rent.movie.category" />
        </rule>
    </profile>
    ...
</TPersonalization>
```

This specification describes how the rule affects all profiles and has medium priority. The information that we need to acquire is the category of the last rented movie. This type of acquisition is implicit and is stored when the member rents a movie in the attribute of the User Model "session.member.lastcategory".

Personalization Rule (1p): MODELLING AND GATHERING PROCESS

Figure 6 shows the part of the NAD diagram in which the rule that models this requirement is introduced. When the link View Recommendations is activated the rule is triggered.

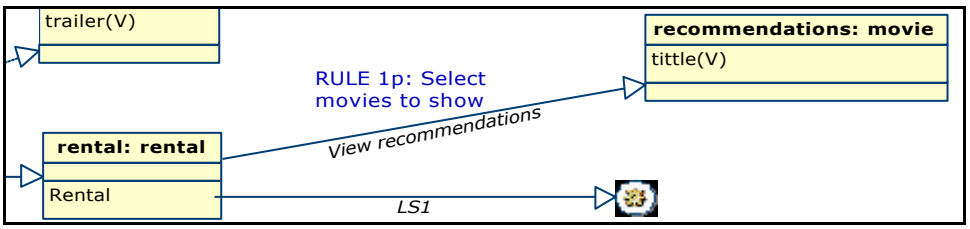

Figure 6 Personalization: rule 1p

Personalization Rule (1p): XML SPECIFICATION

In this case, we have content personalization, because we have to show or hide an specific information. We have as parameter the category of the last movie rented, that was obtained by the acquisition rule. The personalization event indicates that the link "View Recommendations" must be active. If these conditions are accomplished, the action will be executed: the movies with the imposed conditions are shown.

```
<TPersonalization>
    ...
    <profile name="ooh:all" condition="">
       <rule type="personalization:content" name="Recommendations" support="20"
              confidence="100" priority="Medium" activation="12/02/03" expires="12/02/04"
              lastTimeExecuted="23/09/03">
           <params>
           <param name="catmovie" value="session.member.lastcategory"/>
           </params>
           <event type="Navigation" link="View Recommendations"/>
           <condition value="movie.category=catmovie"/>
           <action type="Select" value="movie.category"/>
        </rule>
    </profile>
    ...
</TPersonalization>
```

3. CONCLUSIONS AND FURTHER WORK

OO-H has been extended to support dynamic personalization. OO-H adds acquisition and personalization rules. The needed information to personalize is acquired by means of acquisition rules. To specify the personalization of the application it's needed to define personalization rules. It has been added a rule engine in the OO-H architecture to support the treatment of these rules at execution time. In this way, navigation and presentation models can be compiled to obtain an XML specification that represents the desired dynamic personalization. The final web application is viewed as a composition of a stable and variable part, where the variable part is interpreted by the rule engine to give personalization support. The main benefit is that personalization specification can be modified without need for recompiling the rest of the application modules. Others relevant contributions of this paper are the following: *a User model* that allows to make independent the part personalizable from the stable part from the Domain model, *a Reference model* that provides the needed functionality to support the acquisition and personalization of the application. As further work we have to check the consistency and termination of the rules and study the way of including in OO-H standards of privacy and interchange of user profile information like CRM, CPEX or P3P. Also we want to extend the OO-H Reference model with new type of information to support better personalization of applications.

ACKNOWLEDGEMENT

This paper has been partially supported by the Spanish Ministry of Science and Technology, project number TIC2001-3530-C02-02.

REFERENCES

[1] A. Ginige and S. Murugesan. Web Engineering: an Introduction. IEEE Multimedia Special Issue on Web Engineering, pages 14–18, 04 2001.

[2] S. Ceri, P. Fraternali, and A. Bongio. Web Modeling Language (WebML): a modeling language for designing Web sites WWW9 Conference. In First ICSE Workshop on Web Engineering, International Conference on Software Engineering, 05 2000.

[3] I. Garrigós, J. Gómez and C. Cachero. Modelling Dynamic Personalization in Web Applications. 'Third International Conference on Web Engineering (ICWE'03), LNCS 2722, pages 472-475. Springer-Verlag Berlin Heidelberg, 07 2003.

[4] J. Gómez, C. Cachero, and O. Pastor. Conceptual Modelling of Device-Independent Web Applications. IEEE Multimedia Special Issue on Web Engineering, pages 26–39, 04 2001.

[5] N. Koch, A. Kraus, and R. Hennicker. The Authoring Process of the UML-based Web Engineering Approach. In Proceedings of the 1st International Workshop on Web-Oriented Software Technology, 05 2001.